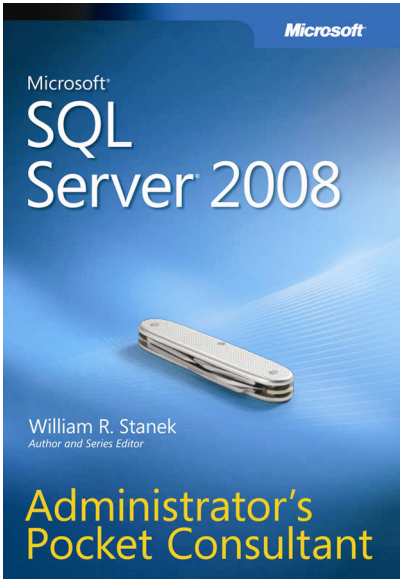

Microsoft® SQL Server® 2008 Administrator's Pocket Consultant

William R. Stanek

To learn more about this book, visit Microsoft Learning at http://www.microsoft.com/MSPress/books/12755.aspx

9780735625891

Microsoft® Press

Table of Contents

Acknowledgments . . . . . *xvii*
Introduction . . . . . *xix*
Who Is This Book For? . . . . . *xix*
How Is This Book Organized? . . . . . *xx*
Conventions Used in This Book . . . . . *xxi*
Support . . . . . *xxii*

Part I Microsoft SQL Server 2008 Administration Fundamentals

1 Microsoft SQL Server 2008 Administration Overview 2
SQL Server 2008 and Your Hardware 3
Microsoft SQL Server 2008 Editions 5
SQL Server and Windows 9
Services for SQL Server 9
SQL Server Logins and Authentication 10
Service Accounts for SQL Server 10
Using the Graphical Administration Tools 12
Using the Command-Line Tools 16
BCP 16
SQLCMD 16
Other Command-Line Tools 18
Using the SQL Server PowerShell 20
Running and Using Cmdlets 20
Running and Using the SQL Server PowerShell 21
Working with SQL Server Cmdlets 22

2 Deploying Microsoft SQL Server 2008 24
SQL Server Integration Roles 24
Using SQL Server Integration Services 24
Using SQL Server 2008 for Relational Data Warehousing 25

Using SQL Server 2008 for Multidimensional Databases and Data Mining 27
Using SQL Server 2008 for Managed Reporting 29
Planning for Your SQL Server 2008 Deployment 30
Building the Server System for Performance 30
Configuring the I/O Subsystem 31
Ensuring Availability and Scalability 33
Ensuring Connectivity and Data Access 35
Managing SQL Server Configuration and Security 36
Running and Modifying SQL Server Setup 37
Creating New Instances of SQL Server 38
Adding Components and Instances. 49
Repairing a SQL Server 2008 Installation 50
Upgrading Your Edition of SQL Server 2008 50
Uninstalling SQL Server 50

3 Managing the Surface Security, Access, and Network Configuration 52

Managing SQL Server Component Feature Access 53
Configuring SQL Server Services 56
Managing the Services Configuration. 57
Managing Service State and Start Mode 61
Setting the Startup Service Account 62
Configuring File Streaming 64
Configuring Service Dump Directories, Error Reporting, and Customer Feedback Reporting 67
Managing the Network and SQL Native Client Configuration 68
Managing the Connections Configuration. 69
Specifying the Shared Memory Network Configuration 71
Specifying the Named Pipes Network Configuration. 71
Specifying the TCP/IP Network Configuration. 72
Configuring Security for Native Client Configurations. 75
Configuring the Native Client Protocol Order 75
Configuring the Shared Memory Native Client Configuration 76
Configuring the Named Pipes Native Client Configuration 76
Configuring the TCP/IP Native Client Configuration 77

4 Configuring and Tuning Microsoft SQL Server 2008 78
Accessing SQL Server Configuration Data 79
Working with the System Catalog and Catalog Views 80
Working with System Stored Procedures 85
Techniques for Managing SQL Server Configuration Options 91
Setting Configuration Options 91
Working with SET Options 92
Working with Server Options 95
Working with Database Options 97
Managing Database Compatibility 98
Configuring SQL Server with Stored Procedures 99
Using SQL Server Management Studio for Queries 99
Executing Queries and Changing Settings 101
Checking and Setting Configuration Parameters 103
Changing Settings with ALTER DATABASE 106

Part II Microsoft SQL Server 2008 Administration

5 Managing the Enterprise 112
Using SQL Server Management Studio 112
Getting Started with SQL Server Management Studio 113
Connecting to a Specific Server Instance 114
Connecting to a Specific Database 115
Managing SQL Server Groups 116
Introducing SQL Server Groups 116
Creating a Server Group 117
Deleting a Server Group 118
Editing and Moving Server Groups 119
Adding SQL Servers to a Group 119
Managing Servers 119
Registering a Connected Server 120
Registering a New Server in the Registered Servers View 121
Registering Previously Registered SQL Server 2000 Servers 122
Updating Registration for Local Servers 122
Copying Server Groups and Registration Details from One Computer to Another 123
Editing Registration Properties 125
Connecting to a Server 125
Disconnecting from a Server 126

Moving a Server to a New Group 126
Deleting a Server Registration 126
Starting, Stopping, and Configuring SQL Server Agent 126
Starting, Stopping, and Configuring Microsoft Distributed Transaction Coordinator 127
Managing SQL Server Startup 128
Enabling or Preventing Automatic SQL Server Startup 128
Setting Database Engine Startup Parameters 129
Managing Services from the Command Line 133
Managing the SQL Server Command-Line Executable File 133
Managing Server Activity 134
Examining Process Information 135
Tracking Resource Waits and Blocks 137
Troubleshooting Deadlocks and Blocking Connections 140
Tracking Command Execution in SQL Server 142
Killing Server Processes 143

6 Configuring Microsoft SQL Server with SQL Server Management Studio 144

Managing the Configuration with SQL Server Management Studio 145
Determining System and Server Information 146
Configuring Authentication and Auditing 147
Setting the Authentication Mode 148
Setting the Auditing Level 148
Enabling or Disabling C2 Audit Logging 148
Enabling or Disabling Common Criteria Compliance 149
Tuning Memory Usage 150
Working with Dynamically Configured Memory 152
Using Fixed Memory 152
Enabling AWE Memory Support 153
Optimizing Memory for Indexing 154
Allocating Memory for Queries 155
Configuring Processors and Parallel Processing 156
Optimizing CPU Usage 156
Setting Parallel Processing 158
Configuring Threading, Priority, and Fibers 159
Configuring User and Remote Connections 161
Setting Maximum User Connections 161
Setting Default Connection Options 162
Configuring Remote Server Connections 164

Managing Server Settings 165
Enabling or Disabling File Streaming Support 166
Setting the Default Language for SQL Server. 166
Allowing and Disallowing Nested Triggers 167
Controlling Query Execution. 167
Configuring Year 2000 Support 168
Managing Database Settings. 168
Setting the Index Fill. 168
Configuring Backup and Restore Time-Out Options 170
Configuring Backup and Restore Retention Options 171
Flushing the Cache with Checkpoints 171
Compressing the Backup Media. 171
Adding and Removing Active Directory Information 172
Troubleshooting Configuration Problems 172
Recovering from a Bad Configuration. 172
Changing Collation and Rebuilding the *master* Database 174

7 Core Database Administration 176

Database Files and Logs 176
Database Administration Basics. 181
Viewing Database Information in SQL Server Management Studio. 181
Viewing Database Information Using T-SQL 183
Checking System and Sample Databases 184
Examining Database Objects. 185
Creating Databases 187
Creating Databases in SQL Server Management Studio. 188
Creating Databases Using T-SQL 192
Altering Databases and Their Options 193
Setting Database Options in SQL Server Management Studio. 193
Modifying Databases Using ALTER DATABASE. 194
Configuring Automatic Options 199
Controlling ANSI Compliance at the Database Level 201
Configuring Parameterization. 203
Configuring Cursor Options 205
Controlling User Access and Database State 206
Setting Online, Offline, or Emergency Mode 208
Managing Cross-Database Chaining and External Access Options 209

Configuring Recovery, Logging, and Disk I/O Error Checking Options . . . 210
Viewing, Changing, and Overriding Database Options . . . 212
Managing Database and Log Size . . . 212
Configuring SQL Server to Automatically Manage File Size . . . 213
Expanding Databases and Logs Manually . . . 213
Compressing and Shrinking a Database Manually . . . 214
Manipulating Databases . . . 218
Renaming a Database . . . 218
Dropping and Deleting a Database . . . 219
Attaching and Detaching Databases . . . 220
Tips and Techniques . . . 223
Copying and Moving Databases . . . 223
Moving Databases . . . 227
Moving and Resizing *tempdb* . . . 228
Creating Secondary Data and Log Files . . . 230
Preventing Transaction Log Errors . . . 230
Preventing a Filegroup Is Full Error . . . 231
Creating a New Database Template . . . 231
Configuring Database Encryption . . . 231

8 Full-Text Search Administration . . . 234

Full-Text Catalogs and Indexes . . . 234
Managing Full-Text Catalogs . . . 238
Viewing Catalog Properties . . . 239
Creating Catalogs . . . 240
Enabling Indexing of Tables and Views . . . 241
Editing Indexing of Tables and Views . . . 244
Disabling and Removing Full-Text Indexing from Tables and Views . . . 244
Populating Full-Text Catalogs . . . 245
Rebuilding Current Catalogs . . . 249
Cleaning Up Old Catalogs . . . 250
Removing Catalogs . . . 250
Managing Full-Text Search . . . 251
Setting the Default Full-Text Language . . . 251
Working with Stoplists . . . 252
Creating Stoplists . . . 253
Managing Stoplists . . . 255
Creating and Using Thesaurus Files . . . 257

9 Managing SQL Server 2008 Security 261
Overview of SQL Server 2008 Security 261
Working with Security Principals and Securables 262
Understanding Permissions of Securables 264
Examining Permissions Granted to Securables 266
SQL Server 2008 Authentication Modes 269
Windows Authentication 270
Mixed Security and SQL Server Logins 270
Special-Purpose Logins and Users 271
Working with the Administrators Group 271
Working with the Administrator User Account 271
Working with the sa Login 272
Working with the NETWORK SERVICE and SYSTEM Logins 272
Working with the Guest User 272
Working with the dbo User 273
Working with the sys and INFORMATION_SCHEMA Users 274
Permissions 274
Object Permissions 274
Statement Permissions 280
Implied Permissions 280
Roles 281
Server Roles 281
Database Roles 282
Managing Server Logins 285
Viewing and Editing Existing Logins 285
Creating Logins 287
Editing Logins with T-SQL 289
Granting or Denying Server Access 291
Enabling, Disabling, and Unlocking Logins 292
Removing Logins 293
Changing Passwords 294
Configuring Server Roles 294
Assigning Roles by Login 294
Assigning Roles to Multiple Logins 296
Revoking Access Rights and Roles by Server Login 297
Controlling Database Access and Administration 297
Assigning Access and Roles by Login 297
Assigning Roles for Multiple Logins 298
Creating Standard Database Roles 299

Creating Application Database Roles 301
Removing Role Memberships for Database Users. 302
Deleting User-Defined Roles. 302
Transact-SQL Commands for Managing Access and Roles 303
Managing Database Permissions 304
Assigning Database Permissions for Statements 305
Object Permissions by Login. 310
Object Permissions for Multiple Logins. 312

Part III Microsoft SQL Server 2008 Data Administration

10 Manipulating Schemas, Tables, Indexes, and Views 316
Working with Schemas. 317
Creating Schemas. 317
Modifying Schemas 319
Moving Objects to a New Schema 320
Dropping Schemas. 322
Getting Started with Tables 322
Table Essentials 323
Understanding Data Pages 323
Understanding Extents. 325
Understanding Table Partitions 325
Working with Tables 326
Creating Tables 326
Modifying Existing Tables 332
Viewing Table Row and Size Information 335
Displaying Table Properties and Permissions 336
Displaying Current Values in Tables. 336
Copying Tables 337
Renaming and Deleting Tables. 337
Adding and Removing Columns in a Table 338
Scripting Tables. 339
Managing Table Values 339
Using Native Data Types 339
Using Fixed-Length, Variable-Length, and Max-Length Fields 343
Using User-Defined Data Types 344
Allowing and Disallowing Nulls 347
Using Default Values 347
Using Sparse Columns 348

Using Identities and Globally Unique Identifiers 348
Using User-Defined Table Types 351
Using Views 354
Working with Views 354
Creating Views 356
Modifying Views 359
Using Updatable Views 360
Managing Views 360
Creating and Managing Indexes 361
Understanding Indexes 361
Using Clustered Indexes 363
Using Nonclustered Indexes 364
Using XML Indexes 364
Using Filtered Indexes 364
Determining Which Columns Should Be Indexed 365
Indexing Computed Columns and Views 367
Viewing Index Properties 367
Creating Indexes 369
Managing Indexes 374
Using the Database Engine Tuning Advisor 377
Column Constraints and Rules 382
Using Constraints 382
Using Rules 386
Creating Partitioned Tables and Indexes 387
Creating Partition Functions 387
Creating Partition Schemes 388
Creating Partitions 389
Viewing and Managing Partitions 390
Compressing Tables, Indexes, and Partitions 392
Using Row and Page Compression 392
Setting or Changing Compression Settings 393

11 Importing, Exporting, and Transforming Data 395

Working with Integration Services 395
Getting Started with Integration Services 396
Integration Services Tools 397
Integration Services and Data Providers 399
Integration Services Packages 399
Creating Packages with the SQL Server Import And Export Wizard 400
Stage 1: Source and Destination Configuration 401

Stage 2: Copy or Query 409
Stage 3: Formatting and Transformation 413
Stage 4: Save and Execute 416
Understanding BCP 419
BCP Basics 420
BCP Syntax 420
BCP Permissions and Modes 423
Importing Data with BCP 423
Exporting Data with BCP 425
BCP Scripts 426
Using the BULK INSERT Command 427

12 Linked Servers and Distributed Transactions 429
Working with Linked Servers and Distributed Data 429
Using Distributed Queries 430
Using Distributed Transactions 433
Running the Distributed Transaction Coordinator Service 434
Managing Linked Servers 435
Adding Linked Servers 435
Configuring Security for Linked Servers 440
Setting Server Options for Remote and Linked Servers 442
Deleting Linked Servers 444

13 Implementing Snapshot, Merge, and Transactional Replication 445
An Overview of Replication 445
Replication Components 446
Replication Agents and Jobs 447
Replication Variants 449
Planning for Replication 451
Replication Models 451
Preliminary Replication Tasks 452
Distributor Administration 456
Setting Up a New Distributor 456
Updating Distributors 460
Creating Distribution Databases 462
Enabling and Updating Publishers 463
Enabling Publication Databases 464
Deleting Distribution Databases 464
Disabling Publishing and Distribution 465

Creating and Managing Publications 465
Creating Publications 465
Viewing and Updating Publications 474
Setting Publication Properties 474
Setting Agent Security and Process Accounts 476
Controlling Subscription Access to a Publication 476
Creating a Script for a Publication 477
Deleting a Publication 478
Subscribing to a Publication 478
Subscription Essentials 478
Creating Subscriptions 479
Viewing Subscription Properties 484
Updating, Maintaining, and Deleting Subscriptions 484
Validating Subscriptions 485
Reinitializing Subscriptions 485

Part IV Microsoft SQL Server 2008 Optimization and Maintenance

14 Profiling and Monitoring Microsoft SQL Server 2008 488
Monitoring Server Performance and Activity 488
Reasons to Monitor SQL Server 488
Getting Ready to Monitor 489
Monitoring Tools and Resources 490
Working with Replication Monitor 492
Starting and Using Replication Monitor 493
Adding Publishers and Publisher Groups 494
Working with the Event Logs 495
Examining the Application Log 497
Examining the SQL Server Event Logs 499
Examining the SQL Server Agent Event Logs 500
Monitoring SQL Server Performance 502
Choosing Counters to Monitor 502
Performance Logging 505
Viewing Data Collector Reports 509
Configuring Performance Counter Alerts 510
Configuring a Management Data Warehouse 511
Understanding Management Data Warehouses 511
Creating the Management Data Warehouse 512
Setting Up Data Collection 512
Managing Collection and Generating Reports 512

Solving Performance Problems with Profiler 513
Using Profiler 513
Creating New Traces 514
Working with Traces 517
Saving a Trace 518
Replaying a Trace 518

15 Backing Up and Recovering Microsoft SQL Server 2008 522

Creating a Backup and Recovery Plan 522
Initial Backup and Recovery Planning 523
Planning for Mirroring and Mirrored Database Backups 527
Planning for Backups of Replicated Databases 528
Planning for Backups of Very Large Databases 529
Planning for Backup Compression 530
Selecting Backup Devices and Media 531
Using Backup Strategies 533
Creating a Backup Device 535
Performing Backups 537
Creating Backups in SQL Server Management Studio 537
Using Striped Backups with Multiple Devices 542
Using Transact-SQL Backup 543
Performing Transaction Log Backups 546
Restoring a Database 548
Database Corruption and Problem Resolution 549
Restoring a Database from a Normal Backup 550
Restoring Files and Filegroups 556
Restoring a Database to a Different Location 558
Recovering Missing Data 559
Creating Standby Servers 559
Using Transact-SQL Restore Commands 561
Restoring the *master* Database 566

16 Database Automation and Maintenance 568

Overview of Database Automation and Maintenance 568
Using Database Mail 569
Performing the Initial Database Mail Configuration 570
Managing Database Mail Profiles and Accounts 575
Viewing or Changing Database Mail System Parameters 577
Using SQL Server Agent 577
Accessing Alerts, Operators, and Jobs 577
Configuring the SQL Server Agent Service 578

Setting the SQL Server Agent Mail Profile 579
Using SQL Server Agent to Restart Services Automatically 579
Managing Alerts 580
Using Default Alerts 580
Creating Error Message Alerts 580
Handling Alert Responses 582
Deleting, Enabling, and Disabling Alerts 583
Managing Operators 584
Registering Operators 584
Deleting and Disabling Notification for Operators 585
Configuring a Fail-Safe Operator 585
Scheduling Jobs 586
Creating Jobs 586
Assigning or Changing Job Definitions 587
Setting Steps to Execute 588
Configuring Job Schedules 592
Handling Job Alerts 595
Handling Notification Messages 595
Managing Existing Jobs 596
Managing Job Categories 597
Automating Routine Server-to-Server Administration Tasks 598
Copying Users, Tables, Views, and Other Objects from One Database to Another 598
Copying Alerts, Operators, and Scheduled Jobs from One Server to Another 601
Multiserver Administration 601
Event Forwarding 602
Multiserver Job Scheduling 603
Database Maintenance 605
Database Maintenance Checklist 605
Using Maintenance Plans 606
Checking and Maintaining Database Integrity 613

17 Managing Log Shipping and Database Mirroring 618

Log Shipping 618
Log Shipping: How It Works 618
Preparing for Log Shipping 620
Upgrading SQL Server 2000 Log Shipping to SQL Server 2008 Log Shipping 621
Enabling Log Shipping on the Primary Database 622

Adding Log Shipping Secondary Databases 626
Changing the Transaction Log Backup Interval 629
Changing the Copy and Restore Intervals 629
Monitoring Log Shipping 630
Failing Over to a Secondary Database 630
Disabling and Removing Log Shipping 633
Database Mirroring 633
Database Mirroring Essentials 634
Configuring Database Mirroring 635
Managing and Monitoring Mirroring 640
Recovering by Using Failover 643
Removing Database Mirroring 644
Using Mirroring and Log Shipping 645

18 Implementing Policy-Based Management 647
Introducing Policy-Based Management 647
Working with Policy-Based Management 649
Managing Policies Throughout the Enterprise 656
Importing and Exporting Policies 656
Configuring Central Management Servers 658
Executing Statements Against Multiple Servers 662
Configuring and Managing Policy Facets 662
Creating and Managing Policy Conditions 664
Creating and Managing Policies 667
Managing Policy Categories and Mandating Policies 669
Evaluating Policies 671
Troubleshooting Policy-Based Management Policies 674

Index 677

Chapter 14

Profiling and Monitoring Microsoft SQL Server 2008

In this chapter:

Monitoring Server Performance and Activity 488
Working with Replication Monitor 492
Working with the Event Logs 495
Monitoring SQL Server Performance 502
Configuring a Management Data Warehouse 511
Solving Performance Problems with Profiler 513

Monitoring server performance, tracking user activity, and troubleshooting errors are essential parts of database administration, and Microsoft SQL Server has several tools that you can use to perform these tasks. Reliability And Performance Monitor, the standard Microsoft Windows Server 2008 tool for monitoring servers, has updated counters for SQL Server. These counters allow you to track many different server resources and activities. SQL Server Profiler, an analysis and profiling tool, allows you to trace server events. Other tools and resources are available, such as stored procedures and the SQL Server logs.

Monitoring Server Performance and Activity

Monitoring SQL Server is not something you should do haphazardly. You need to have a plan—a set of goals that you hope to achieve. Let's look at some reasons you might want to monitor SQL Server and the tools you can use to do this.

Reasons to Monitor SQL Server

One of the main reasons you monitor SQL Server performance is to troubleshoot problems. For example, if users are having problems connecting to the server, you will want to monitor the server to find out more about what is causing these problems. Your goal is to track down the problem using the available monitoring resources and then solve the problem effectively.

Another common reason to monitor SQL Server is to improve server performance. To achieve optimal performance, you need to minimize the time it takes for users to see

the results of queries and maximize the total number of queries that the server can handle simultaneously. You do this by using the following techniques:

- Resolve hardware issues that might be causing problems. For example, if disk read/write activity is slower than expected, work on improving disk input/output (I/O).
- Monitor memory and CPU usage, and take appropriate steps to reduce the load on the server. For example, other processes running on the server might be using memory and CPU resources needed by SQL Server.
- Cut down the network traffic load on the server. With replication, for example, you can configure remote stored procedure execution rather than transmit large data changes individually.

Unfortunately, you often have to make tradeoffs in resource usage. For example, as the number of users accessing SQL Server grows, you might not be able to reduce the network traffic load, but you might be able to improve server performance by optimizing queries or indexing.

Getting Ready to Monitor

Before you start monitoring SQL Server, it is a good idea to establish baseline performance metrics for your server. To do this, you measure server performance at various times and under different load conditions. You can then compare the baseline performance with subsequent performance to determine how SQL Server is performing. Performance metrics that are well above the baseline measurements might indicate areas in which the server needs to be optimized or reconfigured.

After you establish the baseline metrics, prepare a monitoring plan. A comprehensive monitoring plan involves the following steps:

1. Determine which server events should be monitored to help you accomplish your goal.
2. Set filters to preferentially select the amount of information that is collected.
3. Configure monitors and alerts to watch the events.
4. Log the event data so that it can be analyzed.
5. Analyze the event data, and replay the data to find a solution.

These procedures will be examined later in this chapter in "Monitoring SQL Server Performance" on page 502. Although you should develop a monitoring plan in most cases, sometimes you might not want to go through all these steps to monitor SQL Server. For example, if you want to check only current user activity levels, you might not want to use Reliability And Performance Monitor and you might decide to run the stored procedure sp_who instead. Or you can examine this information in the Activity Monitor window in SQL Server Management Studio.

Note The stored procedure sp_who reports on current users and processes. When you execute sp_who, you can pass a login name as an argument. If you do not specify a login name, NULL is passed in this argument, so all logins are returned. If you use the keyword *active* as the login name, you will see only active processes; any processes waiting for the next command from a user will be excluded. Instead of a specific login name, such as *sa*, you can use the numeric value for a system process ID as well.

Monitoring Tools and Resources

The primary monitoring tools you will use are Reliability And Performance Monitor and SQL Server Profiler. Other resources for monitoring SQL Server include

- **Activity Monitor** This monitor provides information on current users, processes, and locks, as discussed in "Managing Server Activity" on page 134. To display Activity Monitor, use the Object Explorer view to access an instance of the Database Engine. Right-click the Database Engine instance and then select Activity Monitor.
- **Replication Monitor** This monitor provides details on the status of SQL Server replication and allows you to configure replication alerts. To display Replication Monitor, use the Object Explorer view to access an instance of the Database Engine. Right-click the Replication node and then select Launch Replication Monitor.
- **SQL Server logs** The information in these event logs allows you to view informational, auditing, warning, and error messages that can help you troubleshoot SQL Server problems. To access the server logs, use the Object Explorer view to access an instance of the Database Engine. Expand the server node and the Management node. Under the Management node, expand the SQL Server Logs node and then double-click the log you want to examine.
- **Job Activity Monitor** This monitor provides details on the status of SQL Server Agent jobs. To display Job Activity Monitor, use the Object Explorer view to access an instance of the Database Engine. Expand the server node and the SQL Server Agent node, and then double-click Job Activity Monitor.
- **SQL Server Agent logs** The information in these event logs allows you to view informational, auditing, warning, and error messages that can help you troubleshoot SQL Server Agent problems. To access agent logs, use the Object Explorer view to access an instance of the Database Engine. Expand the server node and the SQL Server Agent node. Under the SQL Server Agent node, expand the Error Logs node and then double-click the log you want to examine.

> **Note** SQL Server documentation refers to the SQL Server and SQL Server Agent logs as *error logs*. In their current implementation, however, the logs are more accurately called *event logs*, which is the terminology used in this chapter. Similar to event logs in Microsoft Windows, these logs in SQL Server contain informational and security messages as well as error messages.

- **Event logs** The information in the event logs allows you to troubleshoot systemwide problems, including SQL Server and SQL Server Agent problems. To access event logs, click Start, click Administrative Tools, and then select Event Viewer.
- **DBCC statements** This set of commands allows you to check SQL Server statistics, to trace activity, and to check database integrity.
- **sp_helpdb** This stored procedure displays information about databases.
- **sp_helpindex** This stored procedure reports information about indexes on a table or view.
- **sp_helpserver** This stored procedure provides information in SQL Server instances configured for remote access or replication.
- **sp_monitor** This stored procedure shows key SQL Server usage statistics, such as CPU idle time and CPU usage.
- **sp_spaceused** This stored procedure shows an estimate of disk space used by a table, indexed view, or Service Broker queue in the current database.
- **sp_who** This stored procedure shows a snapshot of current SQL Server users and processes.
- **sys.dm_tran_locks** This dynamic management view shows information about object locks.

> **Note** The sys.dm_tran_locks view replaces the sp_lock stored procedure.

In addition to having the use of log files and Transact-SQL statements, you will find a set of built-in functions that return system information. Table 14-1 provides a summary of key functions and their usages. The values returned by these functions are cumulative from the time SQL Server was last started.

Table 14-1 Built-in Functions for Monitoring SQL Server Performance and Activity

Function	Description	Example
@@connections	Returns the number of connections or attempted connections	select @@connections as 'Total Login Attempts'
@@cpu_busy	Returns CPU processing time in milliseconds for SQL Server activity	select @@cpu_busy as 'CPU Busy', getdate() as 'Since'
@@idle	Returns SQL Server idle time in milliseconds	select @@idle as 'Idle Time', getdate() as 'Since'
@@io_busy	Returns I/O processing time in milliseconds	select @@io_busy as 'IO Time', getdate() as 'Since' for SQL Server
@@pack_received	Returns the number of input packets read from the network by SQL Server	select @@pack_received as 'Packets Received'
@@pack_sent	Returns the number of output packets written to the network by SQL Server	select @@pack_sent as 'Packets Sent'
@@packet_errors	Returns the number of network packet errors for SQL Server connections	select @@packet_errors as 'Packet Errors'
@@timeticks	Returns the number of microseconds per CPU clock tick	select @@timeticks as 'Clock Ticks'
@@total_errors	Returns the number of disk read/write errors encountered by SQL Server	select @@total_errors as 'Total Errors', getdate() as 'Since'
@@total_read	Returns the number of disk reads by SQL Server	select @@total_read as 'Reads', getdate() as 'Since'
@@total_write	Returns the number of disk writes by SQL Server	select @@total_write as 'Writes', getdate() as 'Since'
fn_virtualfilestats	Returns input/output statistics for data and log files	select * from fn_ virtualfilestats(null,null)

Working with Replication Monitor

When you have configured replication as discussed in Chapter 13, "Implementing Snapshot, Merge, and Transactional Replication," you will use Replication Monitor to track the status of replication throughout the enterprise. By default, only the currently selected publisher is displayed in the Replication Monitor window, but you can add

any publishers that you want to monitor and organize them into publisher groups as necessary.

Starting and Using Replication Monitor

To start Replication Monitor, right-click the Replication folder in the Object Explorer view, and then select Launch Replication Monitor. Replication Monitor uses icons to indicate the general status of replication. If any publication has an error status, the error status is indicated by a red circle around an X at all levels within Replication Monitor.

When you select a publisher in the left pane, the right pane shows the replication details for that publisher. By default, this information is refreshed every five seconds, and it can be refreshed immediately by pressing F5. As Figure 14-1 shows, the publisher view has three tabs:

- **Publications** Shows individual entries for each configured publication. An icon indicates the type and status of the publication:
  - A purple book icon with a blue circle in it for snapshot replication
  - A blue book icon with a right-facing green arrow in it for transactional replication
  - A yellow book icon with a left-facing green arrow and a right-facing blue arrow in it for merge replication
  - A red circle around an X for error status

  At a glance, you can also see the number of subscriptions to the publication, the number of subscriptions being synchronized, the current average performance for subscribers, and the current worst performance for subscribers.
- **Subscription Watch List** Shows the status of individual subscriptions by type. Use the first drop-down list to specify the type of subscriptions to display and the second drop-down list to specify whether to display all subscriptions of the specified type or some subset, such as the 25 worst-performing subscriptions. Note the status, such as running, error, and so on; the performance level, such as excellent, good, poor, and so on; and the latency.
- **Agents** Shows the SQL Server Agent jobs common to all publications on the selected publisher. To determine if there are potential replication problems, note the status, last start time, and duration. There might be a problem with jobs that have a status of Never Started and with jobs that have been running for a long time.

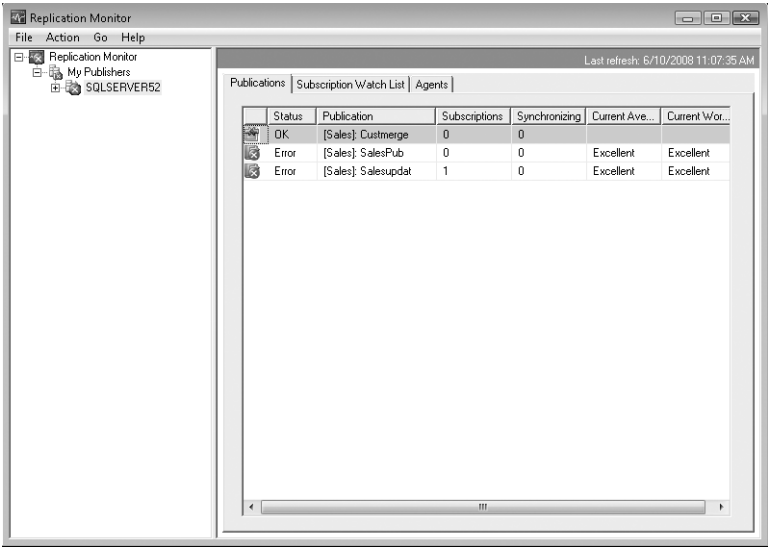

Figure 14-1 Replication Monitor.

Adding Publishers and Publisher Groups

When you first start Replication Monitor, only the currently selected publisher is displayed in Replication Monitor. You can add publishers that you want to monitor and organize them into publisher groups as necessary.

To start monitoring additional publishers and create publisher groups, follow these steps:

1. Start Replication Monitor. In the left pane, right-click the Replication Monitor node and select Add Publisher from the shortcut menu. This displays the Add Publisher dialog box, shown in Figure 14-2.
2. Click Add, and then do the following:
   - Choose Add SQL Server Publisher to configure a connection to a server running SQL Server by using the Connect To Server dialog box. Registered servers are listed in the Server Name drop-down list, and you can browse for others. The default authentication is Windows Authentication, which uses your current login and password. Click Connect.
   - Choose Specify A Distributor And Add Its Publishers to configure a connection to a distributor using the Connect To Server dialog box. Registered servers are listed in the Server Name drop-down list, and you can browse for others. The default authentication is Windows Authentication, which uses your current login and password. When you click Connect, Replication Monitor connects to the distributor, obtains a list of publishers for the distributor, and then connects to these publishers as well.

- Choose Add Oracle Publisher to configure a connection to an Oracle server using the Connect To Server dialog box. Registered servers are listed in the Server Name drop-down list, and you can browse for others. The default authentication is Oracle Standard Authentication, which requires a user login and password. Click Connect.

Note Before you can add an Oracle publisher, you must first configure a connection to the Oracle publisher's distributor by choosing Specify A Distributor And Add Its Publishers.

3. Publisher groups make it easier to manage monitoring in complex enterprise environments. Select the publisher group to which to add the publisher or publishers. If you want to create a new group, click New Group, specify the group name, and then click OK. Then select the new group under Show This Publisher(s) In The Following Group.
4. Click OK.

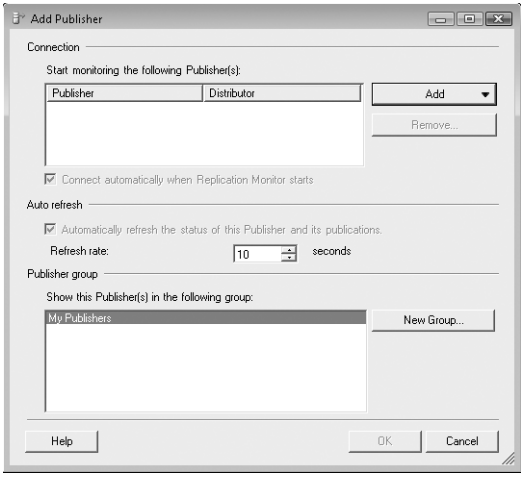

Figure 14-2 The Add Publisher dialog box.

Working with the Event Logs

Event logs provide historical information that can help you track down problems with SQL Server. SQL Server writes events to the SQL Server event logs, the SQL Server Agent event logs, and the Windows application log. You can use all three logs to track messages related to SQL Server. However, there are some things you should know about these logs:

- Only the application log provides additional information on all applications running on the server, and only the application log provides features for filtering

events based on type. For example, you can filter events so that only error and warning messages are displayed.

- If you start the MSSQLServer or MSSQL$*instancename* service from the command prompt, events are logged to the SQL Server event log and to standard output. No events are recorded in the Windows application log.
- Windows has additional logs that can be helpful when tracking issues. If you are tracking security issues, start with the SQL Server event logs and also examine the Windows security log. If you are having trouble finding the source of a problem that is preventing proper operation of SQL Server, start with the SQL Server logs and also examine the Windows application and system logs.

SQL Server error messages can be cryptic and difficult to read if you do not understand the formatting. Error messages logged by SQL Server can have the following information:

- **An error number that uniquely identifies the error message** System error numbers have one to five digits. System errors are numbered from 1 to 50,000. User-defined errors start at 50,001.
- **A severity level that indicates how critical the message is** Severity levels range from 1 to 25. Messages with a severity level of 0 to 10 are informational messages. Severity levels from 11 to 16 are generated by users and users can correct them. Severity levels from 17 to 25 indicate software or hardware errors that you should examine.
- **An error state number that indicates the source of the error** Error state numbers have one to three digits and a maximum value of 127. Normally, error state numbers indicate the line number in the SQL Server code that generated the message.
- **A message that provides a brief description of the error** Read the message to get more information about the error, which will help you in troubleshooting problems.

You might see ODBC (Open Database Connectivity) and OLEDB (object linking and embedding database) return errors from SQL Server that contain similar information as well. The sys.messages catalog view in the *master* database contains a list of error messages and descriptions that can be returned by SQL Server. To see all error messages that can be returned by SQL Server, you can execute the following T-SQL statement:

```
USE master
GO
SELECT * FROM sys.messages
GO
```

Examining the Application Log

The application log contains entries for all database server instances running on the computer, as well as entries for other business applications. You access the application log by completing the following steps:

1. Click Start, click All Programs, click Administrative Tools, and then choose Event Viewer. This starts Event Viewer.
2. Event Viewer displays logs for the local computer by default. If you want to view logs on a remote computer, right-click the Event Viewer node in the console tree (left pane), and then select Connect To Another Computer to display the Select Computer dialog box. In the dialog box, enter the name of the computer you want to access, and then click OK.
3. In the console tree (left pane), expand the Windows Logs node and then click Application. You should see an application log similar to the one shown in Figure 14-3. Use the information in the Source column to determine which service or database server instance logged a particular event.

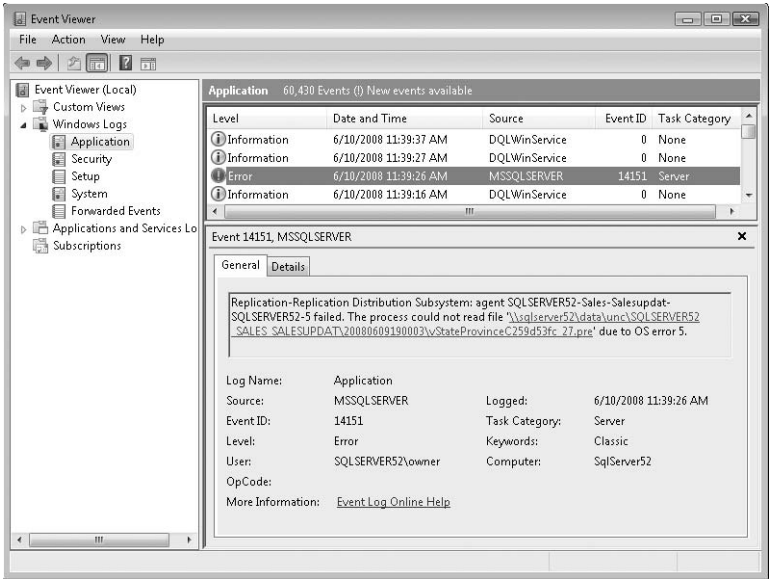

Figure 14-3 A Windows application log.

Note Windows Server 2003 and Windows Server 2008 have different versions of Event Viewer. On Windows Server 2003, Event Viewer doesn't have additional nodes and you can access the application log directly from the console tree.

The entries in the main window of Event Viewer provide a quick overview of when, where, and how an event occurred. To obtain detailed information on an event, review the details provided on the General tab in the lower portion of the main window. The event level or keyword precedes the date and time of the event. Event levels include

- **Information** An informational event that is generally related to a successful action.
- **Audit Success** An event related to the successful execution of an action.
- **Audit Failure** An event related to the failed execution of an action.
- **Warning** A noncritical error that provides a warning. Details for warnings are often useful in preventing future system problems.
- **Error** An error, such as the failure of a service to start.

In addition to the date, time, and event type indicator, the summary and detailed event entries provide the following information:

- **Source** The application, service, or component that logged the event
- **Event ID** An identifier for the specific event
- **Task Category** The category of the event, which is sometimes used to further describe the related action
- **User** The user account that was logged on when the event occurred, if applicable
- **Computer** The computer name on which the event occurred
- **Description** A text description of the event, provided in detailed entries
- **Data** Any data or error code output by the event, provided in detailed entries

Warnings and errors are the two main types of events that you want to examine closely. Whenever one of these types of events occurs and you are unsure of the cause, review the detailed event description. If you want to see only warnings and errors, you can filter the log. To filter a selected log on Windows Server 2003, complete the following steps:

1. From the View menu, choose the Filter option.
2. Clear the following check boxes: Information, Success Audit, and Failure Audit.
3. Select the Warning and Error check boxes if they are not already selected.
4. Click OK. You should now see a list of warning and error messages only. Remember that these messages are for all applications running on the server and not just for SQL Server.

To filter a selected log on Windows Server 2008, complete the following steps:

1. In the actions pane or on the Action menu, click Filter Current Log.
2. Use the Logged list to select the included time frame for logged events. You can choose to include events from the Last Hour, Last 12 Hours, Last 24 Hours, Last 7 Days, or Last 30 Days.
3. Use the Event Level check boxes to specify the level of events to include. Select Verbose to get additional detail.
4. Use the Event Source list to select event sources to include. If you select specific event sources, all other event sources are excluded.
5. Optionally, use the User and Computer(s) boxes to specify users and computers that should be included. If you do not specify the users and computers to be included, events generated by all users and computers are included.
6. Click OK. You should now see a filtered list of events. Review these events carefully, and take steps to correct any problems that exist. To clear the filter and see all events for the log, click Clear Filter in the actions pane or on the Action menu.

Examining the SQL Server Event Logs

The SQL Server logs record information, warnings, errors, and auditing messages pertaining to SQL Server activity. New logs are created when you start the SQL Server service or when you run the sp_cycle_errorlog stored procedure. When a new log is created, the current log is cycled to the archive. SQL Server maintains up to five archived logs (by default).

You can view the SQL Server event logs in SQL Server Management Studio or through a text editor. In SQL Server Management Studio, you access the event logs by completing the following steps:

1. Start SQL Server Management Studio. In the Object Explorer view, connect to the database server of your choice, and then work your way down to the Management folder.
2. Expand the Management folder, and then double-click the SQL Server Logs entry. The current log is shown with the label *Current*. Archived logs are shown with descriptive labels such as *Archive #1*.
3. Double-click the log you want to view to open it in Log File Viewer.
4. With Log File Viewer open, you can add other logs to the log file summary by selecting their check boxes, as shown in Figure 14-4.

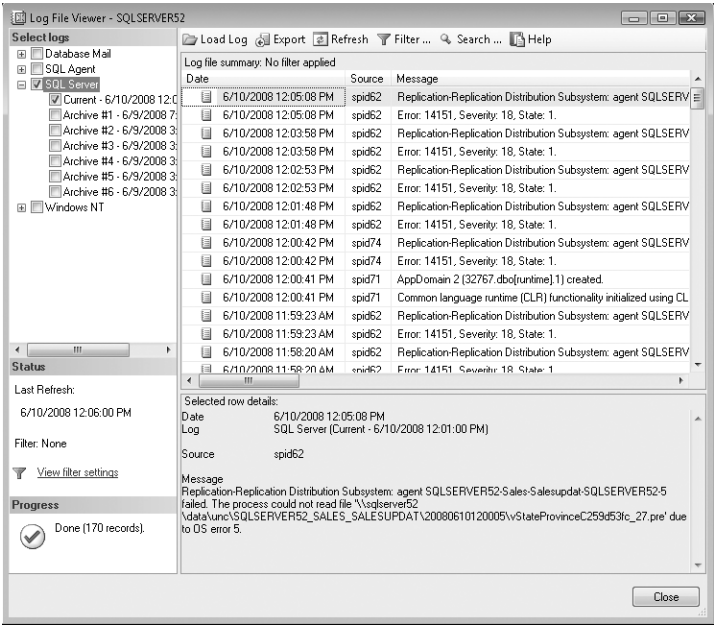

Figure 14-4 Log File Viewer.

To access the event logs in a text editor, complete the following steps:

1. Start a text editor, such as WordPad, and then use its Open dialog box to access the SQL Server Log folder, normally located in MSSQL10.MSSQLSERVER\MSSQL\LOG or MSSQL10.*InstanceName*\MSSQL\LOG.
2. Open the log you want to examine. The current log file is named ERRORLOG with no file extension. The most recent log backup has the extension .1, the second most recent has the extension .2, and so on.

To change the number of logs that SQL Server maintains, right-click the SQL Server Logs entry in the Object Explorer view and select Configure. In the Configure SQL Server Error Logs dialog box, select Limit The Number Of Error Log Files and then set the maximum number of error log files to retain using the Maximum Number Of Error Log Files combo box. The default number of log files maintained is six: one current log and five archive logs. You can change the number of logs maintained to any value from 6 through 99.

Examining the SQL Server Agent Event Logs

The SQL Server Agent logs record information, warnings, and errors pertaining to SQL Server Agent activity. New logs are created only when you start the SQL Server Agent service. When a new log is created, the current log is cycled to the archive. SQL Server maintains up to five archived agent logs (by default).

In SQL Server Management Studio, you access the current SQL Server Agent log by completing the following steps:

1. Start SQL Server Management Studio. In the Object Explorer view, connect to the database server of your choice, and then work your way down to the SQL Server Agent node.
2. Expand the SQL Server Agent node, and then double-click the Error Logs entry. The current log is shown with the label *Current*. Archived logs are labeled *Archive # 1* and so on.
3. Double-click the log you want to view to open it in Log File Viewer.
4. With Log File Viewer open, you can add other logs to the log file summary by selecting their check boxes.

To access archived SQL Server Agent event logs in a text editor, complete the following steps:

1. Start the text editor, and then use its Open dialog box to access the SQL Server Log folder, which is normally located in MSSQL10.MSSQLSERVER\MSSQL\LOG or MSSQL10.*InstanceName*\MSSQL\LOG.
2. Open the log you want to examine. The current log file is named SQLAGENT.OUT. The most recent log backup has the extension .1, the second most recent has the extension .2, and so on.

You can manage the SQL Server Agent logs in several ways. You can force the SQL Server Agent to recycle the current log by right-clicking the SQL Server Agent\Error Logs node in the Object Explorer view, selecting Recycle, and then clicking OK. When you do this, SQL Server closes out the current agent log, moves it to an archive log, and starts a new agent log. You can control the level of logging and set the log file location as well. To do this, complete the following steps:

1. Right-click the SQL Server Agent\Error Logs node in the Object Explorer view, and then select Configure.
2. Use the Error Log File box to set the folder path and file name of the agent log. The default path is MSSQL10.MSSQLSERVER\MSSQL\LOG\SQLAGENT.OUT or MSSQL10.*InstanceName*\MSSQL\LOG\SQLAGENT.OUT. New archive files will also be created in the folder specified as part of the path.
3. Use the Agent Log Level check boxes to control the level of logging for the SQL Server Agent. By default, only error and warning messages are logged. If you want to view informational messages in the logs, select the Information check box as well.
4. Click OK.

Monitoring SQL Server Performance

Reliability And Performance Monitor is the tool of choice for monitoring SQL Server performance. Reliability And Performance Monitor graphically displays statistics for the set of performance parameters you select. These performance parameters are referred to as *counters*.

When you install SQL Server on a system, Reliability And Performance Monitor is updated with a set of counters for tracking SQL Server performance parameters. These counters also can be updated when you install services and add-ons for SQL Server. For example, when you configure replication on a server, Replication Monitor is added and made available through SQL Server Management Studio, and Reliability And Performance Monitor is again updated with a set of objects and counters for tracking replication performance.

Reliability And Performance Monitor creates a graph depicting the various counters you are tracking. You can configure the update interval for this graph, but it's set to three seconds by default. As you will see when you work with Reliability And Performance Monitor, the tracking information is most valuable when you record the information in a log file and when you configure alerts to send messages when certain events occur or when certain thresholds are reached, such as when a database log file gets close to running out of free space.

The following sections examine the procedures you will use with Reliability And Performance Monitor.

> **Note** On Windows Server 2003, you can use Performance Monitor to configure counters to monitor using similar techniques. However, only Reliability And Performance Monitor includes data collector sets for collecting performance data. Data collector sets replace performance monitor logs and alerts.

Choosing Counters to Monitor

Performance Monitor displays information only for counters you are tracking. More than one hundred SQL Server counters are available—and if you have configured other SQL Server features, such as replication, you can use even more counters. These counters are organized into object groupings. For example, all lock-related counters are associated with the SQLServer:Locks object.

To select which counters you want to monitor, complete the following steps:

1. You can access a standalone console by clicking Start, pointing to Administrative Tools, and then clicking Reliability And Performance Monitor. In Server Manager, you can access this tool as a snap-in under the Diagnostics node. Double-click the Diagnostics node to expand it. Finally, double-click and then select the Reliability And Performance node.

2. In the Reliability And Performance console, expand Monitoring Tools and then select Performance Monitor, as shown in Figure 14-5. Any default counters are shown in the lower portion of the Performance Monitor window. To delete a default counter, click its entry in the Performance Monitor window, and then press the Delete key.

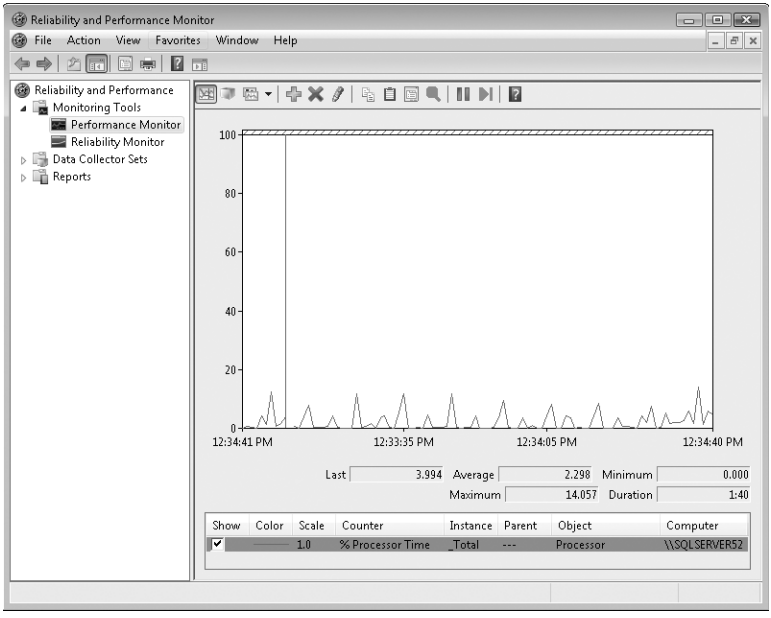

Figure 14-5 The Performance Monitor window.

3. The Performance Monitor tool has several views and view types. Ensure that you are viewing current activity by clicking View Current Activity on the toolbar or pressing Ctrl+T. You can switch between the view types (Line, Histogram Bar, and Report) by clicking Change Graph Type or pressing Ctrl+G.

4. To add counters, click the Add button on the toolbar or press Ctrl+I. This displays the Add Counters dialog box shown in Figure 14-6.

> **Tip** The easiest way to learn what you can track is by exploring the objects and counters available in the Add Counters dialog box. Select an object in the Performance Object list, select the Show Description check box, and then scroll through the list of counters for the object.

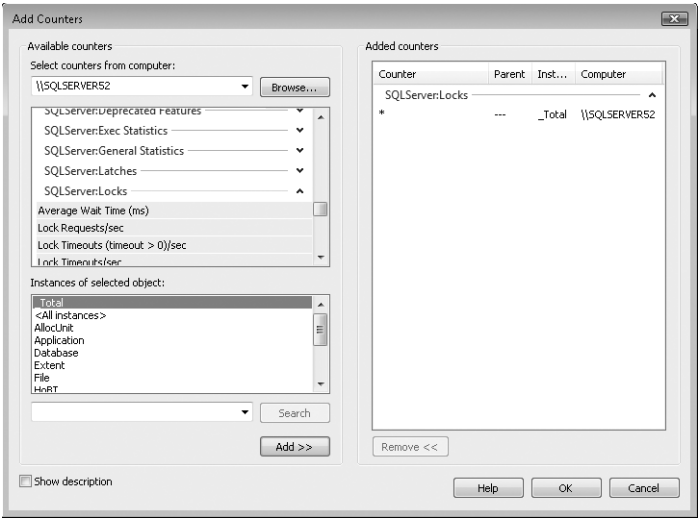

Figure 14-6 The Add Counters dialog box.

5. In the Select Counters From Computer box, enter the Universal Naming Convention (UNC) name of the computer running SQL Server that you want to work with, such as \\SQLServer52, or choose *<Local computer>* to work with the local computer.

   Note You need to be at least a member of the Performance Monitor Users group in the domain or the local computer to perform remote monitoring. When you use performance logging, you need to be at least a member of the Performance Log Users group in the domain or the local computer to work with performance logs on remote computers.

6. In the Available Counters panel, performance objects are listed alphabetically. If you select an object entry by clicking it, all related counters are selected. If you expand an object entry, you can see all the related counters and can then select individual counters by clicking them. For example, you could expand the entry for the SQLServer:Locks object and then select the Average Wait Time (ms), Lock Requests/sec, and Lock Timeouts (timeout > 0)/sec counters.

7. When you select an object or any of its counters, you see the related instances. Choose All Instances to select all counter instances for monitoring. Or select one or more counter instances to monitor. For example, you could select instances of Application or Database locks.

8. When you've selected an object or a group of counters for an object as well as the object instances, click Add to add the counters to the graph. Repeat steps 5 through 8 to add other performance parameters.

9. Click Close when you're finished.

Tip Don't try to chart too many counters or counter instances at once. You'll make the display too difficult to read, and you'll use system resources—namely, CPU time and memory—that might affect server responsiveness.

Performance Logging

Windows Server 2008 introduces data collector sets and reports. Data collector sets allow you to specify sets of performance objects and counters that you want to track. After you've created a data collector set, you can easily start or stop monitoring the performance objects and counters included in the set. In a way, this makes data collector sets similar to the performance logs used in earlier releases of Windows. However, data collector sets are much more sophisticated. You can use a single data set to generate multiple performance counter and trace logs. You can also perform the following tasks:

- Assign access controls to manage who can access collected data.
- Create multiple run schedules and stop conditions for monitoring.
- Use data managers to control the size of collected data and reporting.
- Generate reports based on collected data.

In the Reliability And Performance console, you can review currently configured data collector sets and reports under the Data Collector Sets and Reports nodes, respectively. As shown in Figure 14-7, you'll find data sets and reports that are user defined and system defined. User-defined data sets are created by users for general monitoring and performance tuning. System-defined data sets are created by the operating system to aid in automated diagnostics.

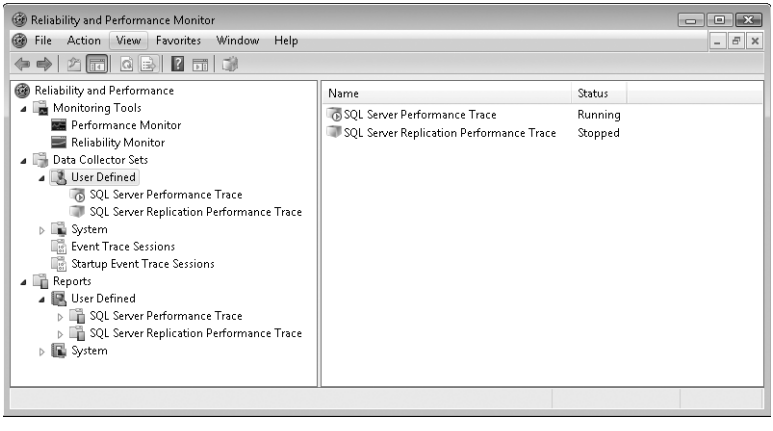

Figure 14-7 Access data collector sets and reports.

Note With SQL Server 2008, you can create several special-purpose data collector sets automatically and then generate periodic reports to get a better understanding of disk usage, query statistics, and server activity. See "Configuring a Management Data Warehouse" on page 511 for details.

Creating and Managing Data Collector Sets

To view the currently configured data collector sets, access Reliability And Performance Monitor by selecting the Reliability And Performance Monitor option on the Administrative Tools menu and then expanding the Data Collector Sets node. You can work with data collectors in a variety of ways:

- You can view currently defined user or system data collector sets by selecting either User Defined or System as appropriate. When you select a data collector set in the left pane, you'll see a list of the related data collectors in the main pane listed by name and type. The Trace type is for data collectors that record performance data whenever related events occur. The Performance Counter type is for data collectors that record data on selected counters when a predetermined interval has elapsed. The Configuration type is for data collectors that record changes to particular registry paths.
- You can view running event traces by selecting Event Trace Sessions. You can then stop a data collector running a trace by right-clicking it and selecting Stop.
- You can view the enabled or disabled status of event traces configured to run automatically when you start the computer by selecting Startup Event Trace Sessions. You can start a trace by right-clicking a startup data collector and selecting Start As Event Trace Session. You can delete a startup data collector by right-clicking it and then selecting Delete.
- You can save a data collector as a template that can be used as the basis of other data collectors by right-clicking the data collector and selecting Save Template. In the Save As dialog box, select a directory, type a name for the template, and then click Save. The data collector template is saved as an XML file that can be copied to other systems.
- You can delete a user-defined data collector by right-clicking it and then selecting Delete. If a data collector is running, you'll need to stop collecting data first and then delete the collector. Deleting a collector deletes the related reports as well.

Collecting Performance Counter Data

Data collectors can be used to record performance data on the selected counters at a specific sample interval. For example, you could sample performance data for the CPU every 15 minutes.

To collect performance counter data, follow these steps:

1. In Reliability And Performance Monitor, under the Data Collector Sets node, right-click the User-Defined node in the left pane, point to New, and then choose Data Collector Set.
2. In the Create New Data Collector Set Wizard, type a name for the data collector, such as SQL Server Performance Monitor or Replication Performance Monitor.
3. Select the Create Manually option and then click Next.
4. On the What Type Of Data Do You Want To Include page, the Create Data Logs option is selected by default. Select the Performance Counter check box and then click Next.
5. On the Which Performance Counters Would You Like To Log page, click Add. This displays the Add Counters dialog box, which you can use as previously discussed to select the performance counters to track. When you are finished selecting counters, click OK.
6. On the Which Performance Counters Would You Like To Log page, type in a sample interval and select a time unit in seconds, minutes, hours, days, or weeks. The sample interval specifies when new data is collected. For example, if you sample every 15 minutes, the data log is updated every 15 minutes. Click Next when you are ready to continue.
7. On the Where Would You Like The Data To Be Saved page, type the root path to use for logging collected data. Alternatively, click Browse and then use the Browse For Folder dialog box to select the logging directory. Click Next when you are ready to continue.

   Best Practices The default location for logging is %*SystemRoot*%\PerfLogs\Admin. Log files can grow in size very quickly. If you plan to log data for an extended period, be sure to place the log file on a drive with lots of free space. Remember, the more frequently you update the log file, the higher the drive space and CPU resource usage on the system.

8. On the Create New Data Collector Set page, the Run As box lists <Default> as the user to indicate that the log will run under the privileges and permissions of the default system account. To run the log with the privileges and permissions of another user, click Change. Type the user name and password for the desired account, and then click OK. User names can be entered in *DOMAIN**Username* format, such as CPANDL\WilliamS for the WilliamS account in the CPANDL domain.
9. Select the Open Properties For This Data Collector Set option and then click Finish. This saves the data collector set, closes the wizard, and then opens the related Properties dialog box.

10. By default, logging is configured to start manually. To configure a logging schedule, click the Schedule tab and then click Add. You can now set the active range, start time, and run days for data collection.
11. By default, logging stops only if you set an expiration date as part of the logging schedule. Using the options on the Stop Condition tab, you can configure the log file to stop manually after a specified period of time, such as seven days, or when the log file is full (if you've set a maximum size limit).
12. Click OK when you've finished setting the logging schedule and stop conditions. You can manage the data collector as explained in "Creating and Managing Data Collector Sets" on page 506.

Note You can configure Windows to run a scheduled task when data collection stops. You configure tasks to run on the Task tab in the Properties dialog box.

Collecting Performance Trace Data

You can use data collectors to record performance trace data whenever events related to their source providers occur. A source provider is an application or operating system service that has traceable events.

To collect performance trace data, follow these steps:

1. In Reliability And Performance Monitor, under the Data Collector Sets node, right-click the User-Defined node in the left pane, point to New, and then choose Data Collector Set.
2. In the Create New Data Collector Set Wizard, type a name for the data collector, such as Database Locks Trace or Database Mirroring Trace.
3. Select the Create Manually option and then click Next.
4. On the What Type Of Data Do You Want To Include page, the Create Data Logs option is selected by default. Select the Event Trace Data check box and then click Next.
5. On the Which Event Trace Providers Would You Like To Enable page, click Add. Select an event trace provider to track, such as Active Directory Domain Services: Core. By selecting individual properties in the Properties list and clicking Edit, you can track particular property values rather than all values for the provider. Repeat this process to select other event trace providers to track. Click Next when you are ready to continue.
6. Complete steps 7 through 12 from the previous procedure, "Collecting Performance Counter Data."

Collecting Configuration Data

You can use data collectors to record changes in registry configuration. To collect configuration data, follow these steps:

1. In Reliability And Performance Monitor, under the Data Collector Sets node, right-click the User-Defined node in the left pane, point to New, and then choose Data Collector Set.
2. In the Create New Data Collector Set Wizard, type a name for the data collector, such as SQL Server Registry or Registry Adapter Info.
3. Select the Create Manually option and then click Next.
4. On the What Type Of Data Do You Want To Include page, the Create Data Logs option is selected by default. Select the System Configuration Information check box and then click Next.
5. On the Which Registry Keys Would You Like To Record page, click Add. Type the registry path to track. Repeat this process to add other registry paths to track. Click Next when you are ready to continue.
6. Complete steps 7 through 12 from the earlier procedure, "Collecting Performance Counter Data."

Viewing Data Collector Reports

When you're troubleshooting problems, you'll often want to log performance data over an extended period of time and then review the data to analyze the results. For each data collector that has been or is currently active, you'll find related data collector reports. As with data collector sets themselves, data collector reports are organized into two general categories: user-defined and system.

You can view data collector reports in Reliability And Performance Monitor. Expand the Reports node and then expand the individual report node for the data collector you want to analyze. Under the data collector's report node, you'll find individual reports for each logging session. A logging session begins when logging starts and ends when logging is stopped.

The most recent log is the one with the highest log number. If a data collector is actively logging, you won't be able to view the most recent log. You can stop collecting data by right-clicking a data collector set and selecting Stop. Collected data is shown by default in a graph view from the start of data collection to the end of data collection.

You can modify the report details using the following techniques:

1. In the monitor pane, press Ctrl+Q or click the Properties button on the toolbar. This displays the Performance Monitor Properties dialog box.
2. Click the Source tab.

3. Specify data sources to analyze. Under Data Source, click Log Files and then click Add to open the Select Log File dialog box. You can now select additional log files to analyze.
4. Specify the time window that you want to analyze. Click Time Range, and then drag the Total Range bar to specify the appropriate starting and ending times. Drag the left edge to the right to move up the start time. Drag the right edge to the left to move down the end time.
5. Click the Data tab. You can now select counters to view. Select a counter and then click Remove to remove it from the graph view. Click Add to display the Add Counters dialog box, which you can use to select the counters that you want to analyze.

   Note Only counters that you selected for logging are available. If you don't see a counter that you want to work with, you'll need to modify the data collector properties, restart the logging process, and then check the logs at a later date.

6. Click OK. In the monitor pane, click the Change Graph Type button to select the type of graphing.

Configuring Performance Counter Alerts

You can configure alerts to notify you when certain events occur or when certain performance thresholds are reached. You can send these alerts as network messages and as events that are logged in the application event log. You can also configure alerts to start application and performance logs.

To configure an alert, follow these steps:

1. In Reliability And Performance Monitor, under the Data Collector Sets node, right-click the User-Defined node in the left pane, point to New, and then choose Data Collector Set.
2. In the Create New Data Collector Set Wizard, type a name for the data collector, such as DB Application Locks Alert or SQL Server Replication Alert.
3. Select the Create Manually option and then click Next.
4. On the What Type Of Data Do You Want To Include page, the Create Data Logs option is selected by default. You don't want to use this option or its related check boxes. Instead, select the Performance Counter Alert option and then click Next.
5. On the Which Performance Counters Would You Like To Monitor page, click Add to display the Add Counters dialog box. This dialog box is identical to the Add Counters dialog box discussed previously. Use the Add Counters dialog box to add counters that trigger the alert. Click OK when you're finished.

6. In the Performance Counters panel, select the first counter and then use the Alert When text box to set the occasion when an alert for this counter is triggered. Alerts can be triggered when the counter is above or below a specific value. Select Above or Below, and then set the trigger value. The unit of measurement is whatever makes sense for the currently selected counter or counters. For example, to alert if processor time is over 95 percent, you would select Over and then type **95**. Repeat this process to configure other counters you've selected.
7. Click Next. Complete steps 8 through 12 from the earlier procedure, "Collecting Performance Counter Data."

Configuring a Management Data Warehouse

SQL Server 2008 has a built-in feature called the management data warehouse. This feature uses several special-purpose data collector sets to automatically collect disk usage, query statistics, and server activity information. To use this feature, you must configure a data collection host and then set up data collection for SQL Server instances you want to track.

Understanding Management Data Warehouses

When you configure a management data warehouse, you can enable data collection whenever you want to monitor SQL Server performance and then generate reports to review the collected information. When you are finished evaluating SQL Server performance, you should free server resources used for collection by disabling data collection.

To enable data collection, you must create a management data warehouse. The warehouse is a database that stores the collected data as well as related report data. As the selected SQL Server instance will then act as a data collection host, you must also ensure that SQL Server Agent is properly configured. SQL Server Agent jobs are used to collected data at periodic intervals on any SQL Server instance for which you've subsequently configured data collection.

Any database you use as a management data warehouse has three special-purpose roles:

- **mdw_reader** Members of this role are able to access the management data warehouse and read reports.
- **mdw_writer** Members of this role are able to write and upload data to the management data warehouse. All data collectors must have this role.
- **mdw_admin** Members of this role have full access and can perform read, write, update, and delete operations in the management data warehouse.

These special-purpose roles are stored in the *msdb* database on the data collection host and no user is a member of these roles by default. While users who are members of the

sysadmin role have full access to data collector views, administrators must be explicitly added to the appropriate role or roles to perform other operations.

Creating the Management Data Warehouse

A management data warehouse stores your data collector information. You can create a management data warehouse by completing the following steps:

1. Start SQL Server Management Studio. In the Object Explorer view, connect to the server you want to use, and then expand the Management folder. If data collection hasn't been configured, you'll see a red down arrow on the Data Collection icon. Right-click Data Collection and then select Configure Management Data Warehouse.
2. Select Create Or Upgrade A Management Data Warehouse and then click Next.
3. SQL Server stores the collected data in a database. If you want to use an existing database to store the data, select the database using the selection list provided. Otherwise, click New and create a database for storing the collected data. Click Next.
4. On the Map Logins And Users page, you can map logins and users to management data warehouse roles. Later, by configuring logins for the *msdb* database, you can modify membership in these roles as discussed in "Managing Server Logins" on page 285. Click Next and then click Finish.

Setting Up Data Collection

You can set up data collection by completing the following steps:

1. In SQL Server Management Studio's Object Explorer view, expand the Management folder, right-click Data Collection, and then select Configure Management Data Warehouse.
2. Select Set Up Data Collection and then click Next.
3. Click the Options (...) button to the right of the Server Name box. Connect to the data collection host. Next, select the collection database using the selection list provided.
4. Optionally set a cache directory that is used to store collected data before it is uploaded to the specified database. If you don't specify a directory, the TEMP directory is used.
5. Click Next and then click Finish.

Managing Collection and Generating Reports

When you set up collection, data collection is enabled automatically. You can turn data collection on or off by right-clicking the Data Collection node and selecting Enable Data Collection or Disable Data Collection as appropriate. In SQL Server Management

Studio's Object Explorer view, you can generate data collection reports by expanding the Management folder, right-clicking Data Collection, pointing to Reports, pointing to Management Data Warehouse, and then selecting the type of report to generate. You can generate the following reports: Server Activity History, Disk Usage History, or Query Statistics History.

Solving Performance Problems with Profiler

Whether you are trying to track user activity, troubleshoot connection problems, or optimize SQL Server, SQL Server Profiler is one of the best utilities available. Profiler enables you to trace events that occur in SQL Server. Events you can track in Profiler are similar to counters you can monitor in Performance Monitor. They are organized into groups called *event classes*, and you can track one or more events for any of the available event classes. The strengths of Profiler are its advanced features and extensive customization capabilities.

You can record and replay Profiler traces when you want to analyze the data—and this is one area in which Profiler excels. You can use Profiler to do the following:

- Use the information to find slow-running queries and then determine what is causing the queries to run slowly.
- Go through statements a step at a time to find the cause of a problem.
- Track a series of statements that cause a particular problem, and then replay the trace on a test server to determine the cause.
- Use trace information to determine the cause of deadlocks.
- Monitor user and application activity to discover actions that are using CPU time or queries that are taking a long time to process.

Let's look at how you can work with Profiler. Then we will examine how to create and manage traces.

Using Profiler

You can start Profiler in two ways:

- Click Start, click All Programs, click Microsoft SQL Server 2008, click Performance Tools, and then choose SQL Server Profiler.
- In SQL Server Management Studio, select SQL Server Profiler from the Tools menu.

Figure 14-8 shows Profiler in the process of running a trace. The columns shown for the trace, such as EventClass, are completely configurable when you are setting up the trace, allowing you to select or clear columns as necessary. Two columns you will want to pay particular attention to are Duration and CPU. The Duration column shows how long a particular event has been running in milliseconds. The CPU column shows the amount of CPU processing time the event requires in milliseconds.

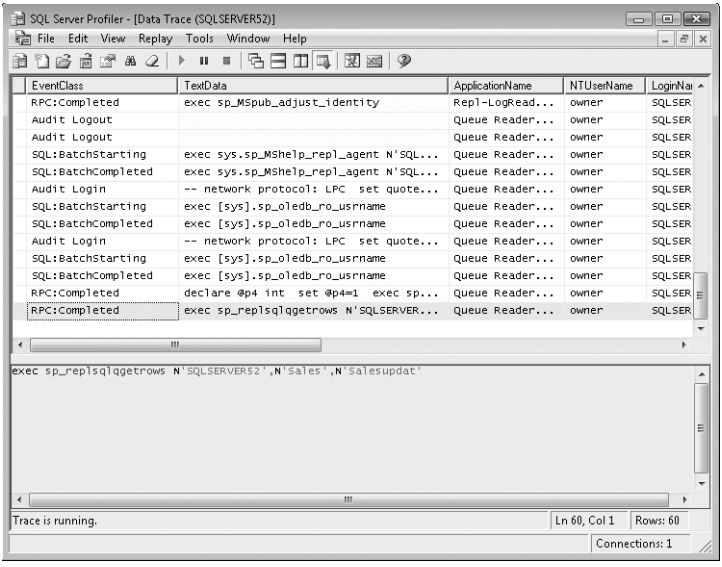

Figure 14-8 Using SQL Server Profiler to trace SQL Server events.

Stored procedures provide an alternative to Profiler. Using these stored procedures gives you some options that you do not have with SQL Server Profiler. You can

- Store traces in the Windows application log.
- Autostart a trace when SQL Server starts.
- Forward event data to another computer running SQL Server (Windows only).

To create traces with stored procedures, complete the following steps:

1. Create a trace definition using sp_trace_create.
2. Set events to capture using sp_trace_setevent.
3. Set event filters using sp_trace_setfilter.

Creating New Traces

You use traces to record events generated by local and remote SQL servers. You run traces in the Profiler window and store them for later analysis.

To start a new trace, complete the following steps:

1. Start SQL Server Profiler, and then click the New Trace button. Or select File, New Trace.
2. Use the Connect To Server dialog box to connect to the server you want to trace.
3. You will see the Trace Properties dialog box, shown in Figure 14-9.

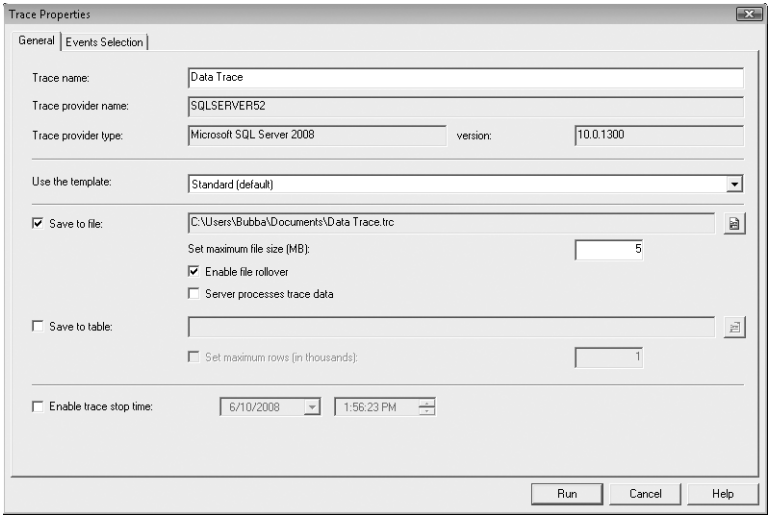

Figure 14-9 The Trace Properties dialog box.

4. In the Trace Name text box, type a name for the trace, such as Data Trace or Deadlock Trace For CustomerDB.

5. You can store traces as they are being created by selecting the Save To File or the Save To Table check box, or both. Or you can store a running trace later by selecting File, selecting Save As, and then choosing either the Trace File option or the Trace Table option.

   Best Practices There are advantages and disadvantages to using trace files and trace tables. You can use trace files to store traces quickly and efficiently using minimal system resources. Trace tables make it easy to store a trace directly in a table on another server, but you use much more of the system resources and usually have slower response times. Note also that storing a trace saves only the trace data. It does not save the trace definition. To reuse the trace definition, you will have to export the trace definition.

6. SQL Profiler templates are used to save trace definitions that contain the events, data columns, and filters used in a trace. The Use The Template drop-down list enables you to choose a template to use as the basis of the trace. Select the TSQL_Replay template if you want to replay the trace.

   Note SQL Profiler templates end with the .tdf file extension.

7. Click the Events Selection tab, as shown in Figure 14-10. The currently selected template determines the events that are selected for tracking by default. The best way to learn the types of events you can trace is to select each event or event class and read its description at the bottom of the Events Selection tab. Move the pointer to a specific column to see details about that data column.

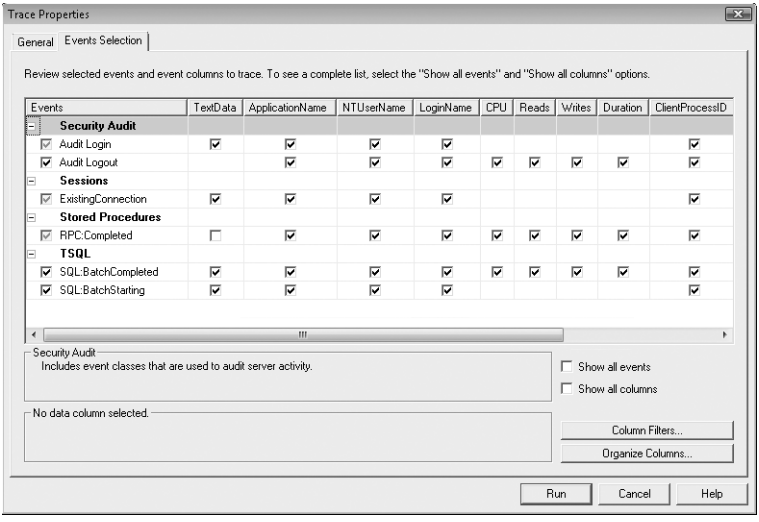

Figure 14-10 Select events to trace.

8. Only a subset of the traceable events and event classes is displayed by default. To see all event classes available, select Show All Events. The event classes that can be traced include Broker, CLR, Cursors, Database, Deprecation, Errors And Warnings, Full Text, Locks, OLEDB, Objects, Performance, Progress Report, Query Notifications, Scans, Security Audit, Server, Sessions, Stored Procedures, TSQL, Transactions, and User Configurable.
9. Only a subset of the traceable data columns is displayed by default. To see all data columns, select Show All Columns.
10. Select event subclasses to add to the trace. If you select a subclass, all data columns for that class are tracked.
11. As necessary, select individual data columns for event subclasses to track specific data columns for an event subclass (versus all data columns for a subclass). At a minimum, you should track
    - Cursors, CursorExecute
    - Cursors, CursorOpen
    - Cursors, CursorPrepare
    - Sessions, ExistingConnection
    - Stored Procedures, RPC:OutputParameter
    - Stored Procedures, RPC:Starting
    - TSQL, Exec Prepared SQL

- TSQL, Prepare SQL
- TSQL, SQL:BatchStarting

Tip If you are tracking distributed queries, be sure to add the HostName column that corresponds to the ServerName in the display window. For transactions, be sure to add the TransactionID column. Also, if you plan to replay the trace for troubleshooting, refer to "Replaying a Trace" on page 518 for specific event classes and data columns that you need to select.

12. To focus the trace on specific types of data, you might want to set criteria that exclude certain types of events. If so, select an event category you want to filter, click the Column Filters button to open the Edit Filter dialog box, and then set filter criteria. For each event category, you can use different filtering criteria. To use the criteria, you click on the related plus sign and then enter the appropriate value in the text box provided. When you are finished, click OK to close the Edit Filter dialog box. You use the filter criteria as follows:

    - **Equals, Not Equal To, Greater Than Or Equal, or Less Than Or Equal** Use these criteria to set the values that trigger the event. Events with values outside the specified range are excluded. For example, with the CPU event category you can specify that only events using greater than or equal to 1000 milliseconds of CPU time are captured. If events use less CPU time than specified, they are excluded.
    - **Like or Not Like** Enter strings to include or exclude for this event category. Use the wildcard character (%) to match a series of characters. Use the semicolon (;) to separate multiple strings. For example, you can use the Application Name category to exclude all application names that start with *MS* and *SQL Server* by typing **MS%;SQL Server%**.

13. When you are finished configuring the trace, click Run to start the trace.

Working with Traces

Profiler displays information for multiple traces in separate windows that can be cascaded or tiled. Use the buttons on the Profiler toolbar to control work with traces. Create a new trace by clicking the New Trace button, and then use the options in the New Trace dialog box to configure the trace. Create a trace template by clicking New Template, setting trace properties, and then clicking Save. Once you have an active trace, you can do the following:

- Start the trace by clicking the Start Selected Trace button.
- Pause the trace by clicking the Pause Selected Trace button. You can then use the Start Selected Trace button to resume the trace at the point at which it was stopped.

- Stop the trace by clicking the Stop Selected Trace button. If you start the trace again with the Start Selected Trace button, the Profiler displays data again from the beginning of the trace process; new data is appended to the files or tables to which you are capturing data.
- Edit trace properties by clicking the Properties button.

Saving a Trace

When you create traces in Profiler, you create trace data and trace definitions. The Profiler window displays trace data, and you can also store it in a file or a table, or both. The trace data records a history of events that you are tracking, and you can use this history to replay the events for later analysis. The Trace Properties dialog box displays the trace definition. You can use the trace definition to create a new trace based on the existing trace.

To save trace data, complete the following steps:

1. Access the Profiler window that displays the trace you want to save.
2. Select File, point to Save As, and then select Trace File or Trace Table.
3. Use the Save As dialog box to select a folder location. Type a file name, and then click Save. Trace files end with the .trc extension.

To save a trace definition, complete the following steps:

1. Access the Profiler window that displays the trace with the definition you want to save.
2. Select File, point to Save As, and then select Trace Template.
3. Use the Select Template Name dialog box to select a folder location. Type a file name, and then click OK. Trace templates end with the .tdf extension.

Replaying a Trace

One of the main reasons you will want to create traces is to save them and replay them later. When replaying traces, Profiler can simulate user connections and authentication, which allows you to reproduce the activity recorded in the trace. You can replay traces in different ways to help you troubleshoot different kinds of problems:

- Execute traces step by step to closely monitor each step in the trace.
- Execute traces using the original timeline to simulate user loads.
- Execute traces with a high replay rate to stress-test servers.

As you monitor the trace execution, you can look for problem areas. Then, when you identify the cause of problems you are trying to solve, you can correct them and rerun the original trace definition. If you are still having problems, you will need to reanalyze the trace data or look at other areas that might be causing problems. Keep in mind that you might need to specify different events to capture in the subsequent trace.

Requirements for Replaying Traces

Traces that you want to replay must contain a minimum set of events and data columns. If the trace does not contain the necessary elements, you will not be able to replay the trace. The required elements are in addition to any other elements that you want to monitor or display with traces.

> **Tip** If you select the TSQL_Replay template, the required event classes and data classes are enabled for tracing by default. If you select another template, you might need to manually select the required event classes and data columns.

You should capture the following event classes to allow a trace to be replayed and analyzed correctly: Audit Login, Audit Logout, ExistingConnection, RPC Output Parameter, RPC:Completed, RPC:Starting, SQL:BatchCompleted, and SQL:BatchStarting. When you are replaying server-side cursors, you must add CursorClose, CursorExecute, CursorOpen, CursorPrepare, and CursorUnprepare. When you are replaying server-side prepared SQL statements, you must add Exec Prepared SQL and Prepare SQL.

You should capture the following data columns to allow a trace to be replayed and analyzed correctly: ApplicationName, Binary Data, ClientProcessID, DatabaseID, DatabaseName, EndTime, Error, EventClass, EventSequence, HostName, IsSystem, LoginName, NTDomainName, NTUserName, ServerName, SPID, StartTime, and TextData.

Replaying Traces on a Different Server

You can replay a trace on a server other than the server originally traced. When you replay a trace on another server, this server is called the *target system*. To replay traces on the target, you should ensure that all logins contained in the trace meet the following criteria:

- Are created on the target system, and are in the same database as the source system
- Have the same permissions they had originally
- Have the same passwords they had originally
- Are set to use a default database that matches the database on the source system

If these settings are not the same, you will see errors, but the replay operation will continue. Also, database IDs on the target system must be the same as those on the source system. The easiest way to set up databases on the target is to complete the following steps:

1. Back up the *master* database on the source and any user databases used in the trace.
2. Restore the databases on the target as explained in "Restoring a Database to a Different Location" on page 558.

Replaying and Analyzing a Trace

Replaying a trace allows you to analyze problems. To begin, start Profiler, and then select the Open Trace File or Open Trace Table option, as appropriate for the type of trace you want to replay. After you select the trace to replay, the trace is then loaded into the Profiler window. Events and commands recorded in the trace are summarized in the Profiler window, as shown in Figure 14-11. You can select an entry to see an expanded list of commands executed.

As Figure 14-11 also shows, the toolbar in the replay window differs from the standard toolbar. The buttons provide just about everything that you need to debug traces, including

- **Start Replay** Starts executing the trace
- **Pause Replay** Pauses execution of the trace
- **Stop Replay** Stops execution of the trace
- **Execute One Step** Allows you to move through the trace one step at a time
- **Run To Cursor** Allows you to move through the trace using cursor sets
- **Toggle Breakpoint** Allows you to set breakpoints for the trace execution

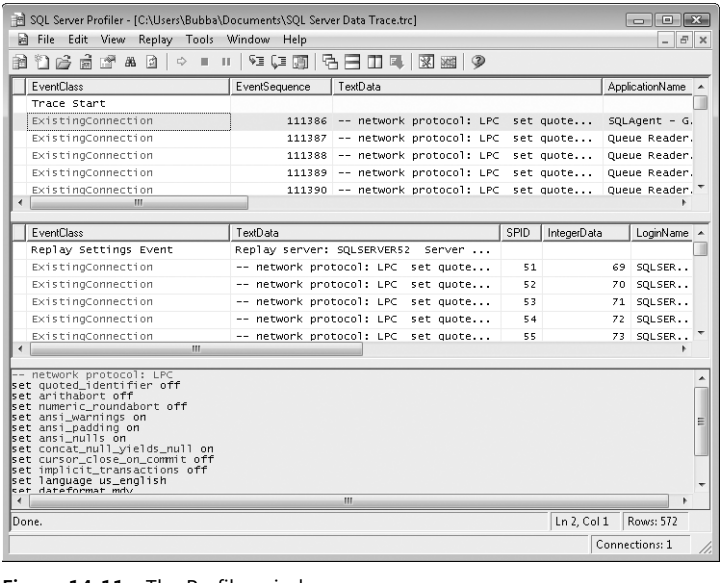

Figure 14-11 The Profiler window.

When you start the replay, you will need to connect to the server, and then the initial dialog box is displayed to configure replay options. (See Figure 14-12.) You configure

the options in the Replay Configuration dialog box to control where and how the playback takes place. Start by setting the destination server for the replay operation. By default, the replay server is set to the current (local) server. Click Change to use a different replay server, and then set the replay options.

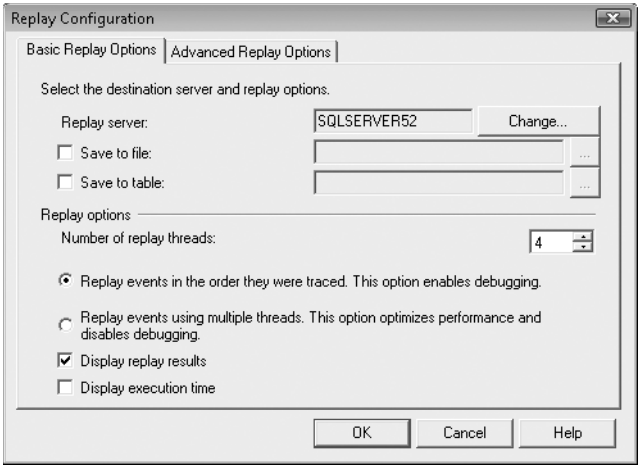

Figure 14-12 The Replay Configuration dialog box.

The replay options determine how closely the replay mirrors the original event execution. You can choose from the following options in the dialog box:

- **Replay Events In The Order They Were Traced** Events are started in the order in which they originally started. This enables debugging, but it does not guarantee timing of event execution. Events might be executed sooner than their original start time or after their original start time, depending on current activity levels, the current speed of connections, and other factors.
- **Replay Events Using Multiple Threads** Events are replayed as quickly as they can be processed. No timing is maintained between events. When one event completes, the next event is started. This optimizes performance and disables debugging.

The Display Replay Results check box controls whether or not the replay results are displayed in the Profiler window. To display results, select this option. Otherwise, clear this option.

You can also select an output file to save the result of the replay for later viewing. The output file allows you to review the replay just as you would any other trace file.